LISTING OF CLAIMS:

1. (Canceled)

2. (Currently Amended) The method of claim [[1]] 12 further comprising:
responsive to receiving the status information, storing the status information.

3. (Currently Amended) The method of claim [[1]] 12, wherein the polling, determining, and initiating steps are performed by a single thread.

4. (Currently Amended) The method of claim [[1]] 3, wherein the single thread is part of a class.

5. (Currently Amended) The method of claim [[1]] 12, wherein the initiating step comprises:
identifying active threads within the plurality of related threads;
identifying inactive threads within the plurality of related threads; and
terminating inactive threads.

6. (Canceled)

7. (Currently Amended) The method of claim [[1]] 12, wherein the plurality of related threads is a plurality of printer threads.

8. (Currently Amended) The method of claim [[1]] 12, wherein the plurality of related threads is a plurality of video threads.

9. (Currently Amended) The method of claim [[1]] 12, wherein the method is implemented in a virtual machine.

10. (Original) The method of claim 9, wherein the virtual machine is a Java virtual machine.

11. (Canceled)

12. (Currently Amended) ~~The method of claim 11,~~ A method in a data processing system for monitoring a plurality of related threads, the method comprising the data processing system implemented steps of:

polling the plurality of related threads for status information;

responsive to receiving the status information, determining whether a thread within a plurality of related threads is inactive; and

responsive to an occurrence of inactivity in a thread within the plurality of related threads in which the inactivity is due to an event, initiating cleanup processes based on the status information, wherein the event is at least one of an occurrence of a period of time or an error.

13. (Canceled)

14. (Canceled)

15. (Currently Amended) The data processing system of claim [[14]] 25 further comprising:

storing means, responsive to receiving the status information, for storing the status information.

16. (Currently Amended) The data processing system of claim [[14]] 25, wherein the polling, determining, and initiating means are preformed by a single thread.

17. (Currently Amended) The data processing system of claim [[14]] 16, wherein the single thread is part of a class.

18. (Currently Amended) The data processing system of claim [[14]] 25, wherein the initiating means comprises:

first identifying means for identifying active threads within the plurality of related threads;

second identifying means for identifying inactive threads within the plurality of related threads; and

terminating means for terminating inactive threads.

19. (Canceled)

20. (Currently Amended) The data processing system of claim [[14]] 25, wherein the plurality of related threads is a plurality of printer threads.

21. (Currently Amended) The data processing system of claim [[14]] 25, wherein the plurality of related threads is a plurality of video threads.

22. (Currently Amended) The data processing system of claim [[14]] 25, wherein the data processing system is implemented in a virtual machine.

23. (Original) The data processing system of claim 22, wherein the virtual machine is a Java virtual machine.

24. (Canceled)

25. (Currently Amended) ~~The data processing system of claim 24,~~ A data processing system for monitoring a plurality of related threads, the data processing system comprising:

polling means for polling the plurality of related threads for status information;

determining means, responsive to receiving the status information, for determining whether a thread within a plurality of related threads is inactive; and

initiating means, responsive to an occurrence of inactivity in a thread within the plurality of related threads in which the inactivity is due to an event, for initiating cleanup processes based on the status information, wherein the event is at least one of an occurrence of a period of time or an error.

26. (Canceled)

27. (Canceled)

28. (Currently Amended) A computer program product in a computer readable medium for monitoring a plurality of related threads, the computer program product comprising:

first instructions for polling the plurality of related threads for status information;

second instructions, responsive to receiving the status information, for determining whether a thread within a plurality of related threads is inactive; and

third instructions, responsive to an occurrence of inactivity in a thread within the plurality of related threads in which the inactivity is due to an event, for initiating cleanup processes based on the status information, wherein the event is at least one of an occurrence of a period of time or an error.

29. (New) The computer program product of claim 28 further comprising:

fourth instructions, responsive to receiving the status information, for storing the status information.

30. (New) The computer program product of claim 28, wherein the polling, determining, and initiating instructions are performed by a single thread.

31. (New) The computer program product of claim 30, wherein the single thread is part of a class.

32. (New) The computer program product of claim 28, wherein the initiating instructions comprise:

first instructions for identifying active threads within the plurality of related threads;

second instructions for identifying inactive threads within the plurality of related threads; and

third instructions for terminating inactive threads.

33. (New) The computer program product of claim 28, wherein the plurality of related threads is a plurality of printer threads.

34. (New) The computer program product of claim 28, wherein the plurality of related threads is a plurality of video threads.

35. (New) The computer program product of claim 28, wherein the computer program product is implemented in a virtual machine.

36. (New) The computer program product of claim 35, wherein the virtual machine is a Java virtual machine.